Quiz Tool

Final Report for CS39440 Major Project

Author: Mr. Michal Goly (mwg2@aber.ac.uk)
Supervisor: Mr. Chris Loftus (cwl@aber.ac.uk)

26th April 2017
Version: 1.0 (Draft)

This report was submitted as partial fulfilment of a BEng degree in
Software Engineering (G600)

Department of Computer Science
Aberystwyth University
Aberystwyth
Ceredigion
SY23 3DB
Wales, UK

Declaration of originality

I confirm that:

- This submission is my own work, except where clearly indicated.
- I understand that there are severe penalties for Unacceptable Academic Practice, which can lead to loss of marks or even the withholding of a degree.
- I have read the regulations on Unacceptable Academic Practice from the University's Academic Quality and Records Office (AQRO) and the relevant sections of the current Student Handbook of the Department of Computer Science.
- In submitting this work I understand and agree to abide by the University's regulations governing these issues.

Name: Michal Goly

Date: 26th April 2018

Consent to share this work

By including my name below, I hereby agree to this dissertation being made available to other students and academic staff of the Aberystwyth Computer Science Department.

Name: Michal Goly

Date: 26th April 2018

Acknowledgements

I am grateful to coffee.

Abstract

Student engagement and live knowledge monitoring are vital in the provision of good quality lecture content in 2018. It is important to make sure audience understands concepts presented, and quizzes can help lecturers judge students' understanding in real time.

Aberystwyth University currently uses Qwizdom[1] live polling tool during the provision of some lectures and practical sessions. The university operates under a single license forcing lecturers to book sessions before they can use the tool. Due to human nature, session hijacking occasionally occurs to the bemusement of both students and lecturers. For example, students could be shown biology slides half way through their geography lecture.

This project focused on the design and development of an in-house built Quiz Tool, enabling multiple lecturers to use it at the same time and potentially making Qwizdom redundant in the future. This ambition could only be achieved if the project was of high quality and its future maintainability was considered at all stages of the design and development.

The Quiz Tool allows lecturers to login using their Google Single Sign-on[2] credentials, upload their PDF lecture slides and create *Lectures* in the system. Each *Lecture* can be then edited, and eligible slides can be marked as quizes. True/false, single and multi choice style quizes are supported. Once a lecturer is happy with their *Lecture*, he can broadcast it and receive a session key which can be shared with students. Lecture slides will be shown to all students and the lecture can be delivered in a traditional fashion up to the moment a slide has been marked as a quiz. Students will then be able to answer the question and polling results will be presented in real time to the lecturer. Lecture sessions broadcasted in the past are kept, and students' answers can be exported as a PDF report for future analysis.

The tool is composed of a back-end with an associated database, and two front-ends. One for lecturers and one for students. Finally, the tool has been successfully developed using an agile methodology, adjusted for a single person project.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Background & Objectives

1.1 Background

1.1.1 Motivation

I enjoy programming, and this project seemed to involve a lot of coding. I was also excited to build a complex, real time system, while applying my previous software engineering experience and learning new technologies at the same time. I knew I would be given a lot of freedom to choose the most appropriate tools for the task, and incorporate modern software development practices, including continuous integration and containerised environments, to deliver good quality software. I was also keen on developing something that could be actually useful to the university once I leave Aberystwyth. It is more motivating to develop a product, while knowing it could be potentially used in real life, as opposed to being forgotten after the submission. I have experienced being presented with wrong slides during a lecture in my first year at university, so I was happy to address the problem of a single session Qwizdom license with my tool.

1.1.2 Technology Considerations

1.1.2.1 Programming Languages

The initial proposal was to create the classroom quiz system using Java[3] to run it natively on Android[4] mobile devices. The lecturer was supposed the create quizzes on his teaching machine, by interacting with the system using a web front end. Lecture slides were then supposed to be broadcasted to his audience, and they could use their mobile phones to answers questions, which would be then sent back to the lecturer for analysis. I have had previous experience with native Android development, therefore this approach seemed like a reasonable option. The only problem was, iOS[5] devices are very popular in the United Kingdom, and developing an app for Android would exclude a good percentage of students from being able to actively participate in lectures presented. I have therefore started to think about alternative approaches.

The second possibility was to use React Native[6], a JavaScript[7] framework allowing developers to create mobile applications in JavaScript and compile it down to both iOS and Android. This approach would still require the web front end for the lecturer to be developed, and a natural choice would be to use React[8] to keep the learning curve as low as possible.

The final alternative considered, was to develop the whole tool as a web application. This way both the front end for lecturers and students could be developed using the same framework. Members of the audience could participate in lectures by accessing the web application using web browsers installed both on their mobile phones, regardless of the operating system, and their laptops. I considered both React and Angular 4[9], since I have already briefly used it before. React is a library for developing user interface, whereas Angular is a web development framework. The only caveat with using Angular is that the developer needs to learn TypeScript[10], which compiles down to JavaScript.

1.1.2.2 The WebSocket Protocol

The Quiz Tool was supposed to allow a lecturer to broadcast his slides to all the students participating in a lecture. This requires a bidirectional communication protocol between the server and the clients. For example, if a lecturer emits the next slide of his presentation, this change should be pushed to the back end, and then the back end needs to be able to send the new slide to all the students. Students' clients should also be able to push quiz answers back to the back end, so they can be presented to the lecturer in some form.

The WebSocket Protocol can be used to create such real time systems. It enables two-way communication between a client and a remote host, making it ideal for instant messaging, gaming applications, and the Quiz Tool. It uses a single TCP connection for trafic in both directions[11].

1.1.2.3 Prototyping

I have followed various tutorials to quickly prototype proof of concept applications, in order to learn more about the technologies which could be useful during the Quiz Tool development. I started with the Socket.io chat tutorial. Socket.io is a JavaScript library enabling bidirectional event-based communication, and uses the WebSocket Protocol internally[12]. I have created a chat application following one of the tutorials on their homepage[13]. I have also learned the basics of Angular by following the Tour of Heroes tutorial[14], and finally tried to understand how Angular could work together with Socket.io by following the MEAN Socket.io tutorial[15].

1.1.3 On-site vs External Hosting

The initial plan to handle authentication in Quiz Tool was to use the university LDAP[16]. This way, lecturers would be able to provide their university credentials, which would be checked using the Bind operation against the university directory of users, to check if a given person is authorised to use the tool. This would require the Quiz Tool to be deployed to a production environment running within the university intranet. The alternative was to deploy the application to an external cloud hosting.

1.1.4 Similar Tools

Qwizdom[1], is the tool currently used by the university to embed quizzes into presentations to judge students' understanding of the content presented. It has to be installed on lecturer's machine as it is a desktop application. Qwizdom integrates with Microsoft PowerPoint[17], by adding an

extra toolbar allowing the lecturer to insert quiz questions into his presentations and then broadcast them. Once a presentation is started, a session key appears which can be shared with the audience. They can then use the key to join the lecture and answer questions once they become available.

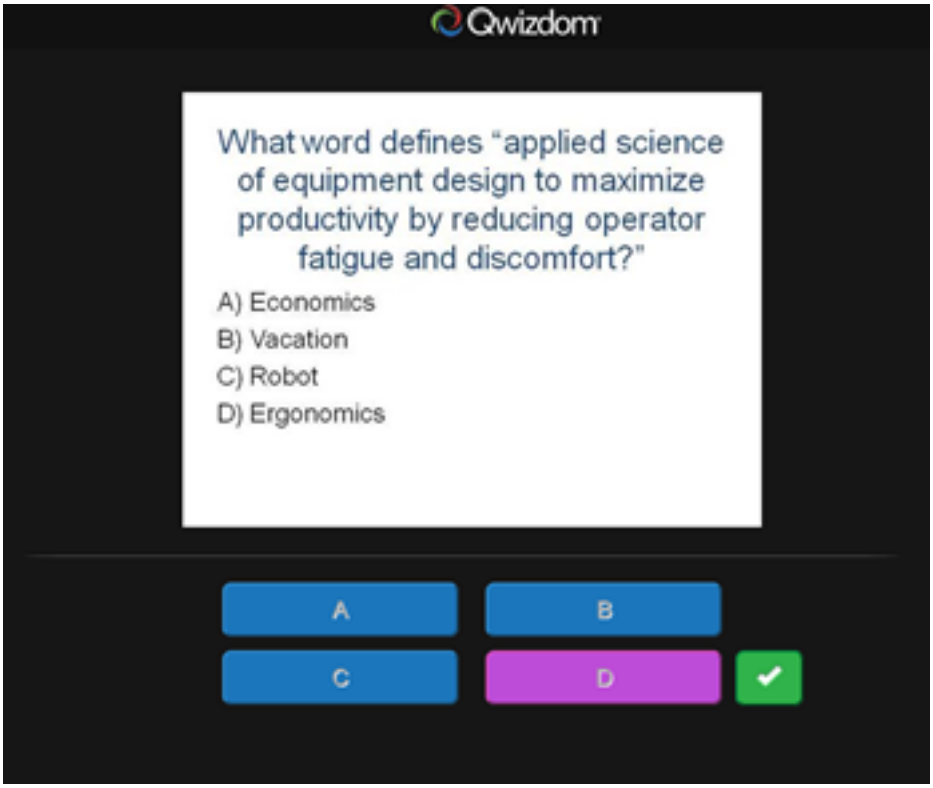

Figure 1.1: Qwizdom web view for the audience

1.2 Analysis

1.2.1 MEAN Stack

Having considered various methods of developing the Quiz Tool, I have decided to choose the web application approach. Allowing students to access the tool using both their mobile devices and laptops, combined with the relatively low learning curve both for me, and for anyone maintaining the software in the future, made it the best option in my opinion. Furthermore, I have decided to develop the application entirely in JavaScript based technologies using the MEAN stack[18].

MEAN stack consists of four elements:

- **M**ongoDB is a JSON document storage NoSQL database
- **E**xpress is a minimalistic JavaScript web development framework
- **A**ngular is a front end web development framework
- **N**odeJS is a JavaScript engine

Front ends for both lecturers and students would be written in Angular, NodeJS would be used on the back end together with Express, and the persistence layer would be provided by MongoDB.

1.2.2 Docker

Each part of the application would be containerised using Docker[19], and containers would run together in the same fashion on the developer's machine, and the build engine using docker-compose[20] container orchestration tool. Docker is an open source engine which can be used to wrap an application and all its dependencies into a lightweight container that can run on any machine capable of running containers[21]. Docker-compose on the other hand, is a tool for defining and running multi-container Docker applications.

1.2.3 AWS Production Environment

Unfortunately, the on-site hosting provided by the university to students to deploy their final projects to was inadequate for the DevOps infrastructure I wanted to put in place. I wanted to have a machine capable of running docker-compose, and some form of continuous integration agent. The LXC debian containers[22] were not compatible with either docker-compose, or Jenkins[23]. The Quiz Tool would be therefore deployed to the production environment provided by an external cloud provider AWS[24], and an alternative build agent would be used. The GitHub Student Developer Pack[25] offers $150 worth of credits for the Amazon cloud, which is why AWS was chosen.

1.2.4 Build

1.2.5 Authentication and Security

1.2.6 Major problems

1.2.7 Top level requirements

1.3 Process

1.3.1 Development Methodology

1.3.2 Version Control

Chapter 2

Design

You should concentrate on the more important aspects of the design. It is essential that an overview is presented before going into detail. As well as describing the design adopted it must also explain what other designs were considered and why they were rejected.

The design should describe what you expected to do, and might also explain areas that you had to revise after some investigation.

Typically, for an object-oriented design, the discussion will focus on the choice of objects and classes and the allocation of methods to classes. The use made of reusable components should be described and their source referenced. Particularly important decisions concerning data structures usually affect the architecture of a system and so should be described here.

How much material you include on detailed design and implementation will depend very much on the nature of the project. It should not be padded out. Think about the significant aspects of your system. For example, describe the design of the user interface if it is a critical aspect of your system, or provide detail about methods and data structures that are not trivial. Do not spend time on long lists of trivial items and repetitive descriptions. If in doubt about what is appropriate, speak to your supervisor.

You should also identify any support tools that you used. You should discuss your choice of implementation tools - programming language, compilers, database management system, program development environment, etc.

Some example sub-sections may be as follows, but the specific sections are for you to define.

2.1 Overall Architecture

2.2 Some detailed design

2.2.1 Even more detail

2.3 User Interface

2.4 Other relevant sections

Chapter 3

Implementation

The implementation should look at any issues you encountered as you tried to implement your design. During the work, you might have found that elements of your design were unnecessary or overly complex; perhaps third party libraries were available that simplified some of the functions that you intended to implement. If things were easier in some areas, then how did you adapt your project to take account of your findings? It is more likely that things were more complex than you first thought. In particular, were there any problems or difficulties that you found during implementation that you had to address? Did such problems simply delay you or were they more significant? You can conclude this section by reviewing the end of the implementation stage against the planned requirements

Chapter 4

Testing

Detailed descriptions of every test case are definitely not what is required here. What is important is to show that you adopted a sensible strategy that was, in principle, capable of testing the system adequately even if you did not have the time to test the system fully.

Provide information in the body of your report and the appendix to explain the testing that has been performed. How does this testing address the requirements and design for the project?

How comprehensive is the testing within the constraints of the project? Are you testing the normal working behaviour? Are you testing the exceptional behaviour, e.g. error conditions? Are you testing security issues if they are relevant for your project?

Have you tested your system on "real users"? For example, if your system is supposed to solve a problem for a business, then it would be appropriate to present your approach to involve the users in the testing process and to record the results that you obtained. Depending on the level of detail, it is likely that you would put any detailed results in an appendix.

The following sections indicate some areas you might include. Other sections may be more appropriate to your project.

4.1 Overall Approach to Testing

4.2 Automated Testing

4.2.1 Unit Tests

4.2.2 User Interface Testing

4.2.3 Stress Testing

4.2.4 Other types of testing

4.3 Integration Testing

4.4 User Testing

Chapter 5

Evaluation

Examiners expect to find in your dissertation a section addressing such questions as:

- Were the requirements correctly identified?
- Were the design decisions correct?
- Could a more suitable set of tools have been chosen?
- How well did the software meet the needs of those who were expecting to use it?
- How well were any other project aims achieved?
- If you were starting again, what would you do differently?

Other questions can be addressed as appropriate for a project.

Such material is regarded as an important part of the dissertation; it should demonstrate that you are capable not only of carrying out a piece of work but also of thinking critically about how you did it and how you might have done it better. This is seen as an important part of an honours degree.

There will be good things and room for improvement with any project. As you write this section, identify and discuss the parts of the work that went well and also consider ways in which the work could be improved.

In the latter stages of the module, we will discuss the evaluation. That will probably be around week 9, although that differs each year.

Appendices

The appendices are for additional content that is useful to support the discussion in the report. It is material that is not necessarily needed in the body of the report, but its inclusion in the appendices makes it easy to access.

For example, if you have developed a Design Specification document as part of a plan-driven approach for the project, then it would be appropriate to include that document as an appendix. In the body of your report you would highlight the most interesting aspects of the design, referring your reader to the full specification for further detail.

If you have taken an agile approach to developing the project, then you may be less likely to have developed a full requirements specification. Perhaps you use stories to keep track of the functionality and the 'future conversations'. It might not be relevant to include all of those in the body of your report. Instead, you might include those in an appendix.

There is a balance to be struck between what is relevant to include in the body of your report and whether additional supporting evidence is appropriate in the appendices. Speak to your supervisor or the module coordinator if you have questions about this.

Appendix A

Third-Party Code and Libraries

If you have made use of any third party code or software libraries, i.e. any code that you have not designed and written yourself, then you must include this appendix.

As has been said in lectures, it is acceptable and likely that you will make use of third-party code and software libraries. If third party code or libraries are used, your work will build on that to produce notable new work. The key requirement is that we understand what is your original work and what work is based on that of other people.

Therefore, you need to clearly state what you have used and where the original material can be found. Also, if you have made any changes to the original versions, you must explain what you have changed.

As an example, you might include a definition such as:

Apache POI library - The project has been used to read and write Microsoft Excel files (XLS) as part of the interaction with the client's existing system for processing data. Version 3.10-FINAL was used. The library is open source and it is available from the Apache Software Foundation [**?**]. The library is released using the Apache License [**?**]. This library was used without modification.

Appendix B

Ethics Submission

Ethics Application Number: 9563

AU Status
Undergraduate or PG Taught

Your aber.ac.uk email address
mwg2@aber.ac.uk

Full Name
Michal Wojciech Goly

Please enter the name of the person responsible for reviewing your assessment.
Prof. Reyer Zwiggelaar

Please enter the aber.ac.uk email address of the person responsible for reviewing your assessment
rrz@aber.ac.uk

Supervisor or Institute Director of Research Department
cs

Module code (Only enter if you have been asked to do so)
CS39440

Proposed Study Title
MMP - Quiz Tool

Proposed Start Date
29/01/2018

Proposed Completion Date
04/05/2018

Are you conducting a quantitative or qualitative research project?
Mixed Methods

Does your research require external ethical approval under the Health Research Authority?
No

Does your research involve animals?
No

Are you completing this form for your own research?
Yes

Does your research involve human participants?
No

Institute
IMPACS

Please provide a brief summary of your project (150 word max)
A web application allowing lecturers to upload their PDF lecture slides, convert certain slides to quizzes (e.g. a slide with bullet points into a single choice A) B) C) quiz), and then broadcasting them to students to monitor their understanding of lecture content presented.

Where appropriate, do you have consent for the publication, reproduction or use of any unpublished material?
Not applicable

Will appropriate measures be put in place for the secure and confidential storage of data?
Yes

Does the research pose more than minimal and predictable risk to the researcher?
Not applicable

Will you be travelling, as a foreign national, in to any areas that the UK Foreign and Commonwealth Office advise against travel to?
No

Please include any further relevant information for this section here:

If you are to be working alone with vulnerable people or children, you may need a DBS (CRB) check. Tick to confirm that you will ensure you comply with this requirement should you identify that you require one.
Yes

Declaration: Please tick to confirm that you have completed this form to the best of your knowledge and that you will inform your department should the proposal significantly change.
Yes

Please include any further relevant information for this section here:

Appendix C

Code Examples

For some projects, it might be relevant to include some code extracts in an appendix. You are not expected to put all of your code here - the correct place for all of your code is in the technical submission that is made in addition to the Final Report. However, if there are some notable aspects of the code that you discuss, including that in an appendix might be useful to make it easier for your readers to access.

As a general guide, if you are discussing short extracts of code then you are advised to include such code in the body of the report. If there is a longer extract that is relevant, then you might include it as shown in the following section.

Only include code in the appendix if that code is discussed and referred to in the body of the report.

3.1 Random Number Generator

The Bayes Durham Shuffle ensures that the psuedo random numbers used in the simulation are further shuffled, ensuring minimal correlation between subsequent random outputs [**?**].

```
#define IM1 2147483563
#define IM2 2147483399
#define AM (1.0/IM1)
#define IMM1 (IM1-1)
#define IA1 40014
#define IA2 40692
#define IQ1 53668
#define IQ2 52774
#define IR1 12211
#define IR2 3791
#define NTAB 32
#define NDIV (1+IMM1/NTAB)
#define EPS 1.2e-7
#define RNMX (1.0 - EPS)
```

```
double ran2(long *idum)
{
  /*-------------------------------------------------*/
  /* Minimum Standard Random Number Generator        */
  /* Taken from Numerical recipies in C              */
  /* Based on Park and Miller with Bays Durham Shuffle */
  /* Coupled Schrage methods for extra periodicity   */
  /* Always call with negative number to initialise  */
  /*-------------------------------------------------*/

  int j;
  long k;
  static long idum2=123456789;
  static long iy=0;
  static long iv[NTAB];
  double temp;

  if (*idum <=0)
  {
    if (-(*idum) < 1)
    {
      *idum = 1;
    }else
    {
      *idum = -(*idum);
    }
    idum2=(*idum);
    for (j=NTAB+7;j>=0;j--)
    {
      k = (*idum)/IQ1;
      *idum = IA1 *(*idum-k*IQ1) - IR1*k;
      if (*idum < 0)
      {
        *idum += IM1;
      }
      if (j < NTAB)
      {
        iv[j] = *idum;
      }
    }
    iy = iv[0];
  }
  k = (*idum)/IQ1;
  *idum = IA1*(*idum-k*IQ1) - IR1*k;
  if (*idum < 0)
  {
    *idum += IM1;
  }
```

```
  k = (idum2)/IQ2;
  idum2 = IA2*(idum2-k*IQ2) - IR2*k;
  if (idum2 < 0)
  {
    idum2 += IM2;
  }
  j = iy/NDIV;
  iy=iv[j] - idum2;
  iv[j] = *idum;
  if (iy < 1)
  {
    iy += IMM1;
  }
  if ((temp=AM*iy) > RNMX)
  {
    return RNMX;
  }else
  {
    return temp;
  }
}
```

Bibliography

[1] Qwizdom, Qwizdom homepage, 2018. [Online] Available: https://qwizdom.com/, [Accessed: Apr. 9, 2018].

Qwizdom is the tool currently used by the univeristy and will be replaced with this project.

[2] Google Sign-In, "Google Identity homepage", 2018. [Online] Available: https://developers.google.com/identity/, [Accessed: Apr. 10, 2018].

Google Sign-In is a secure authentication system that reduces the burden of login for your users, by enabling them to sign in with their Google account. The same account they already use with Gmail, Play, Google+, and other Google services.

[3] Java Programming Language, "Java homepage", 2018. [Online] Available: https://docs.oracle.com/javase/8/docs/technotes/guides/language/index.html, [Accessed: Apr. 11, 2018].

The Java Programming Language is a general-purpose, concurrent, strongly typed, class-based object-oriented language. It is normally compiled to the bytecode instruction set and binary format defined in the Java Virtual Machine Specification.

[4] Android, "Android homepage", 2018. [Online] Available: https://www.android.com/, [Accessed: Apr. 11, 2018].

Android is a mobile operating system developed by Google.

[5] iOS, "Apple homepage", 2018. [Online] Available: https://www.apple.com/uk/ios/ios-11/, [Accessed: Apr. 11, 2018].

iOS is a mobile operating system created and developed by Apple Inc.

[6] React Native, "React Native homepage", 2018. [Online] Available: https://facebook.github.io/react-native/, [Accessed: Apr. 11, 2018].

React Native lets you build mobile apps for both iOS and Android using only JavaScript.

[7] JavaScript Programming Language, "JavaScript Mozilla docs", 2018. [Online] Available: https://developer.mozilla.org/bm/docs/Web/JavaScript, [Accessed: Apr. 11, 2018].

JavaScript is a lightweight, interpreted, programming language, best known for being the scripting language of the web.

[8] React, "React homepage", 2018. [Online] Available: https://reactjs.org/, [Accessed: Apr. 11, 2018].

A JavaScript library for building user interfaces.

[9] Angular, ”Angular homepage”, 2018. [Online] Available: https://angular.io/, [Accessed: Apr. 11, 2018].

Angular is a TypeScript-based front end development framework supported by Google.

[10] TypeScript Programming Language, ”TypeScript homepage”, 2018. [Online] Available: https://www.typescriptlang.org/, [Accessed: Apr. 11, 2018].

TypeScript is a typed superset of JavaScript that compiles to plain JavaScript.

[11] I. Fette, A. Melnikov, ”The WebSocket Protocol”, [Online] Available: https://tools.ietf.org/html/rfc6455, [Accessed: Apr. 11, 2018].

The WebSocket Protocol enables two-way communication between a client running untrusted code in a controlled environment to a remote host that has opted-in to communications from that code.

[12] Socket.io, ”Socket.io homepage”, 2018. [Online] Available: https://socket.io/, [Accessed: Apr. 11, 2018].

Socket.io is a JavaScript library enabling bidirectional event-based communication.

[13] Chat Application Tutorial, ”Socket.io homepage”, 2018. [Online] Available: https://socket.io/get-started/chat/, [Accessed: Apr. 11, 2018].

A Socket.io tutorial guiding the student in developing a basic chat application using Socket.io.

[14] Tour of Heroes Tutorial, ”Angular homepage”, 2018. [Online] Available: https://angular.io/tutorial, [Accessed: Apr. 11, 2018].

The Tour of Heroes tutorial covers the fundamentals of Angular, by building an app that helps staffing agency manage its stable of heroes.

[15] Building Chat Application using MEAN Stack (Angular 4) and Socket.io, ”djamware.com”, 2018. [Online] Available: https://www.djamware.com/post/58e0d15280aca75cdc948e4e/building-chat-application-using-mean-stack-angular-4-and-socketio, [Accessed: Apr. 11, 2018].

Step by step tutorial of building simple chat application using MEAN stack (Angular 4) and Socket.io.

[16] J. Sermersheim, Ed. Novell, ”Lightweight Directory Access Protocol (LDAP): The Protocol”, [Online] Available: https://tools.ietf.org/html/rfc4511, [Accessed: Apr. 11, 2018].

Aberystwyth University directory of users could be checked using the Bind operation to authenticate lecturers once they provide their email and password.

[17] Microsoft PowerPoint, ”products.office.com”, 2018. [Online] Available: https://products.office.com/en-gb/powerpoint, [Accessed: Apr. 11, 2018].

A proprietary tool owned by Microsoft to create presentations.

[18] MEAN stack, [Online] Available: https://github.com/linnovate/mean, [Accessed: Apr. 11, 2018].

The MEAN stack uses Mongo, Express, Angular(4) and Node for simple and scalable full-stack js applications.

[19] Docker, ”Docker homepage”, [Online] Available: https://www.docker.com/, [Accessed: Apr. 11, 2018].

Docker allows app to be containerised and run on multiple hosts in the same fashion.

[20] docker-compose, ”docker-compose homepage”, [Online] Available: https://docs.docker.com/compose/, [Accessed: Apr. 11, 2018].

Compose is a tool for defining and running multi-container Docker applications.

[21] M. OǴara, ”Ben Golub, Who Sold Gluster to Red Hat, Now Running dotCloud”, 26th July 2013. [Online] Available: http://maureenogara.sys-con.com/node/2747331, [Accessed: Apr. 11, 2018].

A journal article mentioning what Docker does.

[22] LXC, ”LXC”, [Online] Available: https://wiki.debian.org/LXC, [Accessed: Apr. 11, 2018].

Linux Containers (LXC) provide a Free Software virtualization system for computers running GNU/Linux. They are inappropriate for the task at hand, as they do not allow running docker-compose due to the lack of system permissions.

[23] Jenkins, ”Jenkins homepage”, [Online] Available: https://jenkins.io/, [Accessed: Apr. 11, 2018].

Jenkins is an industry proven automation tool. It is typically deployed on premises as a Java web application and gives developers more control over the build agents compared to its competitors.

[24] Amazon Web Services, ”AWS homepage”, [Online] Available: https://aws.amazon.com/, [Accessed: Apr. 11, 2018].

AWS provides cloud computing plaftoms to individuals, companies and governments. The production environment of Quiz Tool is hosted on AWS.

[25] GitHub, ”Student Developer Pack”, [Online] Available: https://education.github.com/pack, [Accessed: Apr. 11, 2018].

GitHub partners with top companies to allow students gain experience with real world tools. It funded the production environment hosting on AWS of the Quiz Tool.